Attack Trees with Sequential Conjunction

Barbara Kordy, Ravi Jhawar, Sjouke Mauw, Sasa Radomirovic, Roland Trujillo-Rasua

Attack Trees with Sequential Conjunction

Ravi Jhawar[1], Barbara Kordy[2], Sjouke Mauw[1], Saša Radomirović[3], Rolando Trujillo-Rasua[1]

[1] University of Luxembourg, SnT, Luxembourg
[2] INSA Rennes, IRISA, France
[3] Inst. of Information Security, Dept. of Computer Science, ETH Zürich, Switzerland

Abstract. We provide the first formal foundation of SAND attack trees which are a popular extension of the well-known attack trees. The SAND attack tree formalism increases the expressivity of attack trees by introducing the sequential conjunctive operator SAND. This operator enables the modeling of ordered events.
We give a semantics to SAND attack trees by interpreting them as sets of series-parallel graphs and propose a complete axiomatization of this semantics. We define normal forms for SAND attack trees and a term rewriting system which allows identification of semantically equivalent trees. Finally, we formalize how to quantitatively analyze SAND attack trees using attributes.

Keywords: Attack trees, security modeling, sequential operators, SAND

1 Introduction

Attack trees allow for an effective security analysis by systematically organizing the different ways in which a system can be attacked into a tree. The root node of an attack tree represents the *attacker's goal* and the children of a given node represent its refinement into *sub-goals*. A refinement is typically either disjunctive (denoted by OR) or conjunctive (denoted by AND). The leaves of an attack tree represent the attacker's actions and are called *basic actions*.

Since their inception by Schneier [19], attack trees have quickly become a popular modeling tool for security analysts. However, the limitations of this formalism, in particular with respect to expressing the order in which the various attack steps are executed, have been recognized by many authors (see e.g., [10]). In practice, modeling of security scenarios often requires constructs where conditions on the execution order of the attack components can be clearly specified. This is for instance the case when the time or (conditional) probability of an attack is considered, as in [2,21]. Consequently, several studies have extended attack trees informally with sequential conjunctive refinements. Such extensions have resulted in improved modeling and analyses (e.g., [21,15,22]) and software tools, e.g., ATSyRA [16].

Even though the sequential conjunctive refinement, that we denote by SAND, is well understood at a conceptual level and even applied to real world scenarios [16], none of the existing solutions have provided a rigorous mathematical

formalization of attack trees with SAND. Indeed, the extensions found in the literature are rather diverse in terms of application domain, interpretation, and formality. Thereby, it is infeasible to answer fundamental questions such as: What is the precise expressibility of SAND attack trees? When do two such trees represent the same security scenario? Or what type of attributes can be synthesized on SAND attack trees in the standard bottom-up way? These questions can only be precisely answered if SAND attack trees are provided with a formal, general, and explicit interpretation, i.e., if they are given a formal foundation.

Contributions: In this article we formalize the meaning of a SAND attack tree by defining its semantics. Our semantics is based on series-parallel (SP) graphs, which is a well-studied branch of graph theory. We provide a complete axiomatization for the SP semantics and show that the SP semantics for SAND attack trees are a conservative extension of the multiset semantics for standard attack trees [13] (i.e., our extension does not introduce unexpected equivalences w.r.t. the multiset semantics). To do so, we define a term rewriting system that is terminating and confluent and obtain normal forms for SAND attack trees. As a consequence, we achieve the rather surprising result that the domains of SAND attack trees and sets of SP graphs are isomorphic. We also extend the notion of attributes for SAND attack trees which enable the quantitative analysis of attack scenarios using the standard bottom-up evaluation algorithm.

Organization: Section 2 summarizes the related work and puts our work in context. Section 3 provides a formal definition of SAND attack trees and its semantics using series-parallel graphs. Section 4 defines a complete set of axioms for SAND attack trees and presents a term rewriting system which allows identification of semantically equivalent SAND attack trees. Section 5 outlines an approach to quantitatively analyze SAND attack trees using attributes. Finally, Section 6 concludes with an outlook on future work.

2 Related Work and Motivation

One of the goals of our work is to provide a level of abstraction that encompasses most of the existing approaches from literature. Several extensions of attack trees with temporal or causal dependencies between attack steps have been proposed. We observe that there are three different approaches to achieve this goal. The first approach is to use standard attack trees with the added assumption that the children of an AND node are sequentially ordered from left to right. The second approach is to introduce a mechanism for ordering events in an attack tree, for instance by adding a new type of edge to express causality or conditionality. In its most general case, any partial order on the events in an attack tree can be specified. The third approach consists of the introduction of a new type of node for sequencing. Most extensions fall in this category. This approach is used by authors who require their formalism to be backward compatible, or who need standard, as well as ordered conjunction. We discuss for each of these approaches the most relevant papers with respect to the present article.

Approaches with a sequential interpretation of `AND`. In their work on *Bayesian networks for security*, Qin and Lee define a transformation from attack trees to Bayesian networks [17]. They state that "there always exists an implicit dependent and sequential relationship between `AND` nodes in an attack tree." Most literature on attack trees seem to contradict this statement, implying that there is a need to explicitly identify such sequential relationships.

Jürgenson and Willemson developed an algorithm to calculate the *expected outcome* of an attack tree [22]. The goal of the algorithm is to determine a permutation of leaves for which the optimal expected outcome for an attacker can be achieved. In essence, their input is an attack tree where an `AND` node represents all possible sequences of its children. A peculiarity of their interpretation is that multiple occurrences of the same node are considered only once, implying that the execution of the same action twice cannot be expressed.

Approaches introducing a general order. Peine et al. introduce *security goal indicator trees* [14] in which nodes can be related by a notion of *conditional dependency* and Boolean connectors. The authors, however, do not formally specify the syntax and semantics of the model. A more general approach is proposed by Piètre-Cambacédès and Bouissou [15], who apply *Boolean logic driven Markov processes* to security modeling. Their formalism does not introduce new gates, but a (trigger-)relation on the nodes of the attack tree. Although triggers can express a more general sequential relation than the `SAND` operator, they lack the readability of standard attack tree operators.

Vulnerability cause graphs [1,3] combine properties of attack trees (`AND` and `OR` nodes) and attack graphs (edges express order rather than refinement). The interaction between the `AND` nodes and the order relation is defined through a graph transformation called *conversion of conjunctions*, which ignores the order between nodes. This discrepancy could be solved by considering distinct conjunctive and sequential conjunctive nodes, as we do in this paper.

Approaches introducing sequential `AND`. As noted by Arnold et al. [2], the analysis of time-dependent attacks requires attack trees to be extended with a sequential operator. This is accomplished by defining sequential nodes as conjunctive nodes with a notion of progress of time. The authors define a formal semantics for this extension based on cumulative distribution functions (CDFs), where a CDF denotes the probability that a successful attack occurs within time t. The main difference with our work is that their approach is based on an explicit notion of time, while we have a more abstract approach based on causality. In their semantics, the meaning of an extended attack tree is a CDF, in which the relation to the individual basic attacks is not explicit anymore. In contrast, in our semantics the individual basic attacks and their causal ordering remain visible. As such, our semantics can be considered more abstract, and indeed, we can formulate their semantics as an *attribute* in our approach.

Enhanced attack trees [4] (EATs) distinguish between `OR`, `AND` and `OAND` (Ordered `AND`). Similarly to the approach of Arnold et al. [2], ordered `AND` nodes are used to express temporal dependencies between attack components. The authors evaluate EATs by transforming them into tree automata. Intermediate states in

the automaton support the task of reporting partial attacks. However, because every intermediate node of the tree corresponds to a state in the tree automaton, their approach does not scale well. This problem can be addressed by considering the normal form of attack trees, as proposed in this article.

Not every extension of attack trees with SAND refinements concerns time-dependent attack scenarios; some aim at supporting risk analyses with conditional probabilities. For that purpose, Wen-Ping and Wei-Min introduce *improved attack trees* [21]. The concepts, however, are described at an intuitive level only.

Unified parameterizable attack trees [20] unify different extensions of attack trees (structural, computational, and hybrid). The authors consider two types of ordered AND connectors: priority-based connectors and time-based connectors. The children of the former are ordered from highest to lowest priority, whereas the children of the latter are ordered temporally. Our formalism gives a single interpretation to the SAND operator, yet it can capture both connectors.

Khand [7] extends attack trees with a set of gates from dynamic fault tree modeling, which includes the priority AND gate. Khand assigns truth values to his attack trees by giving truth tables for all gates. When restricted to AND, OR, and priority AND, the truth tables constitute an attribute which is compatible (in the sense of [9]) with our SP semantics for SAND attack trees.

We observe that the extensions of attack trees with sequential conjunction are rather diverse in terms of application domain, interpretation, and formality. In order to give a clear and unambiguous interpretation of the SAND operator and capture different application domains, it is necessary to give a formal semantics as a translation to a well-understood domain. Note that, neither the multiset [13] nor the propositional semantics [11] can express ordering of attack components. Therefore, a richer semantical domain needs to be defined. The purpose of this article is to address this problem.

3 Attack Trees with Sequential Conjunction

We extend the attack tree formalism so that a refinement of a (sub-)goal of an attacker can be a sequential conjunct (denoted by SAND) in addition to disjuncts and conjuncts. We first give a definition of attack trees with the new sequential operator and then define series-parallel graphs on which the semantics for the new attack trees is based.

3.1 SAND Attack Trees

Let $\mathbb{B}$ denote the set of all possible basic actions of an attacker. We formalize standard attack trees introduced in [19] and call them simply *attack trees* in the rest of this paper. Attack trees are closed terms over the signature $\mathbb{B} \cup \{\mathtt{OR}, \mathtt{AND}\}$, generated by the following grammar, where $b \in \mathbb{B}$ is a terminal symbol.

$$t ::= \quad b \mid \mathtt{OR}(t, \ldots, t) \mid \mathtt{AND}(t, \ldots, t) \tag{1}$$

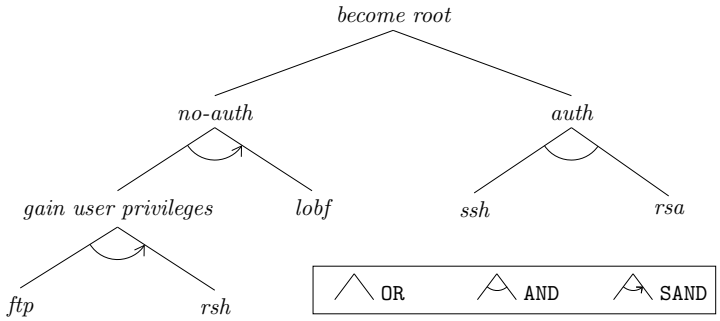

Fig. 1. An attack tree with sequential and parallel conjunctions

The universe of attack trees is denoted by $\mathbb{T}$. SAND *attack trees* are closed terms over the signature $\mathbb{B} \cup \{\mathtt{OR}, \mathtt{AND}, \mathtt{SAND}\}$, where SAND is a non-commutative operator called *sequential conjunction*, and are generated by the grammar

$$t ::= \quad b \mid \mathtt{OR}(t, \ldots, t) \mid \mathtt{AND}(t, \ldots, t) \mid \mathtt{SAND}(t, \ldots, t). \tag{2}$$

The universe of SAND attack trees is denoted by $\mathbb{T}_{\mathtt{SAND}}$. The purpose of OR and AND refinements in SAND attack trees is the same as in attack trees. The sequential conjunctive refinement SAND allows us to model that a certain goal is reached if and only if all its subgoals are reached in a precise order.

The following attack scenario motivates the need for extending attack trees with sequential conjunctive refinement.

Example 1. Consider a file server S, offering ftp, ssh, and rsh services. The attack tree in Figure 1 shows how an attacker can gain root privileges on S (*become root*), in two ways: either without providing any user credentials (*no-auth*) or by breaching the authentication mechanism (*auth*). In the first case, the attacker must first gain user privileges on S (*gain user privileges*) and then perform a local buffer overflow attack (*lobf*). Since the attack steps must be executed in this particular order, the use of SAND refinement is substantial. To gain user privileges, the attacker must exploit an ftp vulnerability to anonymously upload a list of trusted hosts to S (*ftp*).[4] Finally, she can use the new trust condition to remotely execute shell commands on S (*rsh*). The second way is to abuse a buffer overflow in both the ssh daemon (*ssh*) and the RSAREF2 library (*rsa*) used for authentication. These attacks can be executed in any order, which is modeled with the standard AND refinement. Using the term notation introduced in this section, we can represent the SAND attack tree from Figure 1 as

$$t = \mathtt{OR}\big(\mathtt{SAND}\big(\mathtt{SAND}(\mathit{ftp}, \mathit{rsh}), \mathit{lobf}\big), \mathtt{AND}(\mathit{ssh}, \mathit{rsa})\big),$$

where $\mathit{ftp}, \mathit{rsh}, \mathit{lobf}, \mathit{ssh}, \mathit{rsa} \in \mathbb{B}$ are basic actions.

[4] For readability, attack actions are named after the services that are exploited.

3.2 Series-Parallel Graphs

A *series-parallel graph* (SP graph) is an edge-labeled directed graph that has two unique, distinct vertices, called *source* and *sink*, and that can be constructed with the two operators for sequential and parallel composition of graphs that we formally define below. A source is a vertex which has no incoming edges and a sink is a vertex without outgoing edges.

Our formal definition of SP graphs is based on *multisets*, i.e., sets in which members are allowed to occur more than once. We use $\{\!\!\{ \cdot \}\!\!\}$ to denote multisets and $\mathcal{P}(\cdot)$ to denote the powerset of a set or multiset. The *support* $M^\star$ of a multiset M is the set of distinct elements in M. For instance, the support of the multiset $M = \{\!\!\{b_1, b_2, b_2\}\!\!\}$ is $M^\star = \{b_1, b_2\}$. In order to define SP graphs, we first introduce the notion of source-sink graphs labeled by the elements of $\mathbb{B}$.

Definition 1. *A* source-sink graph *over $\mathbb{B}$ is a tuple $G = (V, E, s, z)$, where V is the set of vertices, E is a multiset of labeled edges with support $E^\star \subseteq V \times \mathbb{B} \times V$, $s \in V$ is the unique source, $z \in V$ is the unique sink, and $s \neq z$.*

The sequential composition of a source-sink graph $G = (V, E, s, z)$ with a source-sink graph $G' = (V', E', s', z')$, denoted by $G \cdot G'$, is the graph resulting from taking the disjoint union of G and G' and identifying the sink of G with the source of G'. More precisely, let $\dot\cup$ denote the disjoint union operator and $E^{[s/z]}$ denote the multiset of edges in E, where all occurrences of vertex z are replaced by vertex s. Then we define

$$G \cdot G' = (V \setminus \{z\} \dot\cup V', E^{[s'/z]} \dot\cup E', s, z').$$

The parallel composition, denoted by $G \parallel G'$, is defined similarly, except that the two sources are identified and the two sinks are identified. Formally, we have

$$G \parallel G' = (V \setminus \{s, z\} \dot\cup V', E^{[s'/s, z'/z]} \dot\cup E', s', z').$$

It follows directly from the definitions that the sequential composition is associative and that the parallel composition is associative and commutative.

We write $\xrightarrow{b}$ for the graph with a single edge labeled with b and define SP graphs as follows.

Definition 2. *The set $\mathbb{G}_{\mathcal{SP}}$ of* series-parallel graphs *(SP graphs) over $\mathbb{B}$ is defined inductively by the following two rules*

- *For $b \in \mathbb{B}$, $\xrightarrow{b}$ is an SP graph.*
- *If G and G' are SP graphs, then so are $G \cdot G'$ and $G \parallel G'$.*

It follows directly from Definition 2 that SP graphs are connected and acyclic. Moreover, every vertex of an SP graph lies on a path from the source to the sink. We consider two SP graphs to be *equal* if there is a bijection between their sets of vertices that preserves the edges and edge labels.

Example 2. Figure 2 shows an example of an SP graph with the source s and the sink z. This graph corresponds to the construction

$$\big(\xrightarrow{a} \parallel \xrightarrow{b} \parallel \xrightarrow{b} \big) \cdot \xrightarrow{c} \cdot \Big(\big(\xrightarrow{d} \cdot (\xrightarrow{e} \parallel \xrightarrow{f}) \big) \parallel \xrightarrow{g} \Big).$$

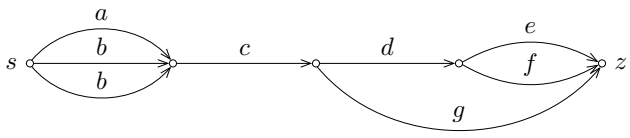

Fig. 2. A series-parallel graph

3.3 SP Semantics for SAND Attack Trees

Numerous semantics have been proposed to interpret attack trees, including propositional logic [12], multisets [13], De Morgan lattices [11], tree automata [4], and Markov processes [15,2]. The choice of a semantics allows us to accurately represent the assumptions made in a security scenario, e.g., whether actions can be repeated or resources reused, and to decide which trees represent the same security scenario. The advantages of formalizing attack trees and the need for various semantics have been discussed in [9]. Since attack trees are AND/OR trees, the most natural interpretation is based on propositional logic. However, because the logical operators are idempotent, this interpretation assumes that the multiplicity of an action is irrelevant. As a consequence, the propositional semantics is not well suited to reason about scenarios with multiple occurrences of the same action. Due to this lack of expressivity a semantics was proposed [13] in which the multiplicity of actions is taken into account. This was achieved by interpreting an attack tree as a set of multisets that represent different ways of reaching the root goal. This multiset semantics is compatible with computations that depend on the number of occurrences of an action in the tree, such as the minimal time to carry out the attack represented by the root goal.

We now extend the multiset semantics to SAND attack trees. Since SP graphs naturally extend multisets with a partial order, they supply a formalism in which we can interpret trees using both commutative and sequential conjunctive refinements. SP graphs therefore provide a canonical semantics for SAND trees in which multiplicity and ordering of goals and actions are significant. The idea is to interpret an attack tree t as a set of SP graphs. The semantics $[\![t]\!]_{\mathcal{SP}} = \{G_1, \ldots, G_k\}$ of a tree t corresponds to the set of possible attacks G_i, where each attack is described by an SP graph labeled by the basic actions of t.

Definition 3. *The* SP semantics *for* SAND *attack trees is given by the function* $[\![\cdot]\!]_{\mathcal{SP}} : \mathbb{T}_{\mathtt{SAND}} \to \mathcal{P}(\mathbb{G}_{\mathcal{SP}})$*, which is defined recursively as follows: for* $b \in \mathbb{B}$*,* $t_i \in \mathbb{T}_{\mathtt{SAND}}$*,* $1 \leq i \leq k$*,*

$$
\begin{aligned}
&[\![b]\!]_{\mathcal{SP}} = \{\overset{b}{\rightarrow}\} \\
&[\![\mathtt{OR}(t_1, \ldots, t_k)]\!]_{\mathcal{SP}} = \textstyle\bigcup_{i=1}^{k} [\![t_i]\!]_{\mathcal{SP}} \\
&[\![\mathtt{AND}(t_1, \ldots, t_k)]\!]_{\mathcal{SP}} = \{G_1 \parallel \ldots \parallel G_k \mid (G_1, ..., G_k) \in [\![t_1]\!]_{\mathcal{SP}} \times ... \times [\![t_k]\!]_{\mathcal{SP}}\} \\
&[\![\mathtt{SAND}(t_1, \ldots, t_k)]\!]_{\mathcal{SP}} = \{G_1 \cdot \ldots \cdot G_k \mid (G_1, ..., G_k) \in [\![t_1]\!]_{\mathcal{SP}} \times ... \times [\![t_k]\!]_{\mathcal{SP}}\}.
\end{aligned}
$$

In the SP semantics, the basic actions of a SAND attack tree are the edges of series-parallel graphs. The semantics of a disjunctive, conjunctive, and sequential conjunctive node are the union, parallel composition, and sequential composition, respectively, of all combinations of SP graphs in the sets that represent the semantics of the node's children.

Example 3. The SP semantics of the attack tree t depicted in Figure 1 is

$$[\![t]\!]_{\mathcal{SP}} = \{\xrightarrow{ftp}\xrightarrow{rsh}\xrightarrow{lobf}\ ,\ \xrightarrow{ssh}\|\xrightarrow{rsa}\}.$$

As shown in Example 3, the SP semantics provides an alternative graph representation for attack trees and therefore contributes a different perspective on an attack scenario. The SAND attack tree emphasizes the refinement of goals, whereas SP graphs highlight the sequential aspect of attacks.

The SP semantics provides a natural partition of $\mathbb{T}_{\texttt{SAND}}$ into equivalence classes.

Definition 4. *Two* SAND *attack trees* t_1 *and* t_2 *are* equivalent with respect to the SP semantics *if and only if they are interpreted by the same set of SP graphs, i.e.,* $[\![t_1]\!]_{\mathcal{SP}} = [\![t_2]\!]_{\mathcal{SP}}$.

By Definition 4, if the SP semantics provides accurate assumptions for an attack scenario, then two SAND attack trees represent the same attack scenario if and only if they are equivalent with respect to the SP semantics.

4 Axiomatization of the SP Semantics

In this section we introduce a complete axiomatization of SAND attack trees with respect to the SP semantics. Such an axiomatization provides us with syntactic transformation rules for SAND attack trees that preserve the trees' SP semantics. In other words, it allows us to manipulate SAND attack trees without the need to convert them to SP graphs. Moreover, we derive a term rewriting system from the axiomatization as a means to effectively decide whether two SAND attack trees are equivalent with respect to the SP semantics. As a consequence, we obtain a canonical representation of SAND attack trees which we prove to be isomorphic to sets of SP graphs.

4.1 A complete set of axioms for the SP semantics

Let $\mathbb{V}$ be a set of variables denoted by capital letters. Following the approach developed in [9], we axiomatize SAND attack trees with equations $l = r$, where l and r are terms over variables in $\mathbb{V}$, constants in $\mathbb{B}$, and the operators AND, OR, and SAND. The equations formalize the intended properties of refinements and provide semantics-preserving transformations of SAND attack trees.

Example 4. Let Sym_ℓ denote the set of all bijections from $\{1,\ldots,\ell\}$ to itself. The axiom $\texttt{AND}(Y_1,\ldots,Y_\ell) = \texttt{AND}(Y_{\sigma(1)},\ldots,Y_{\sigma(\ell)})$, where $\sigma \in \mathrm{Sym}_\ell$, expresses that the order between children refining a parallel conjunctive node is not relevant. In other words, the operator AND is commutative. This implies that any two trees of the form $\texttt{AND}(t_1,\ldots,t_l)$ and $\texttt{AND}(t_{\sigma(1)},\ldots,t_{\sigma(l)})$ represent the same scenario.

Our goal is to define a complete set of axioms, denoted by $E_{\mathcal{SP}}$, for the SP semantics for SAND attack trees. Intuitively, $E_{\mathcal{SP}}$ is a set of equations that can be applied to transform a SAND attack tree into any equivalent SAND attack tree with respect to the SP semantics. Before defining the set $E_{\mathcal{SP}}$, we formalize the notion of a complete set of axioms for a given semantics for (SAND) attack trees, following [9].

Let $T(\mathbb{V},\Sigma)$ be the free term algebra over the set of variables $\mathbb{V}$ and signature Σ, and let E be a set of equations over $T(\mathbb{V},\Sigma)$. The equation $t = t'$, where $t,t' \in T(\mathbb{V},\Sigma)$, is a *syntactic consequence* of E (denoted by $E \vdash t = t'$) if it can be derived from E by application of the following rules. For all $t,t',t'' \in T(\mathbb{V},\Sigma)$, $\rho\colon \mathbb{V} \to T(\mathbb{V},\Sigma)$, and $X \in \mathbb{V}$:

- $E \vdash t = t$,
- if $t = t' \in E$, then $E \vdash t = t'$,
- if $E \vdash t = t'$, then $E \vdash t' = t$,
- if $E \vdash t = t'$ and $E \vdash t' = t''$, then $E \vdash t = t''$.
- if $E \vdash t = t'$, then $E \vdash \rho(t) = \rho(t')$,
- if $E \vdash t = t'$, then $E \vdash t''[t/X] = t''[t'/X]$, where $t''[t/X]$ is the term obtained from t'' by replacing all occurrences of the variable X with t.

Let $\mathbb{T}^{\mathbb{V}}_{\texttt{SAND}}$ denote the set of terms constructed from the set of variables $\mathbb{V}$, the set of basic actions $\mathbb{B}$ (treated as constants), and operators OR, AND and SAND. Let $\mathbb{T}^{\mathbb{V}}$ be the set of terms constructed from the same parts, except for the operator SAND. Using the notion of syntactic consequence, we define a complete set of axioms for a semantics for attack trees.

Definition 5. *Let $[\![\cdot]\!]$ be a semantics for attack trees (resp.* SAND *attack trees) and let E be a set of equations over $\mathbb{T}^{\mathbb{V}}$ (resp. $\mathbb{T}^{\mathbb{V}}_{\texttt{SAND}}$). The set E is a* complete set of axioms *for $[\![\cdot]\!]$ if and only if, for all $t,t' \in \mathbb{T}$ (resp. $\mathbb{T}_{\texttt{SAND}}$)*

$$[\![t]\!] = [\![t']\!] \iff E \vdash t = t'.$$

We are now ready to give a complete set of axioms for the SP semantics for SAND attack trees. These axioms allow us to determine whether two visually distinct trees represent the same security scenario according to the SP semantics.

Theorem 1. *Given $k,m \geq 0$, and $\ell \geq 1$, let $\overline{X} = X_1,\ldots,X_k$, $\overline{Y} = Y_1,\ldots,Y_\ell$, and $\overline{Z} = Z_1,\ldots,Z_m$ be sequences of variables. Let Sym_ℓ be the set of all bijections from $\{1,\ldots,\ell\}$ to itself. The following set of equations over $\mathbb{T}^{\mathbb{V}}_{\texttt{SAND}}$, denoted by $E_{\mathcal{SP}}$, is a complete set of axioms[5] for the SP semantics for* SAND *attack trees.*

$$\texttt{OR}(Y_1,\ldots,Y_\ell) = \texttt{OR}(Y_{\sigma(1)},\ldots,Y_{\sigma(\ell)}), \quad \forall\sigma \in \mathrm{Sym}_\ell \qquad (E_1)$$

[5] Note that the axioms are in fact *axiom schemes*. The operators OR, AND and SAND are *unranked*, representing infinitely many k-ary function symbols ($k \geq 1$).

$$\texttt{AND}(Y_1, \ldots, Y_\ell) = \texttt{AND}(Y_{\sigma(1)}, \ldots, Y_{\sigma(\ell)}), \quad \forall \sigma \in \mathrm{Sym}_\ell \tag{E_2}$$

$$\texttt{OR}\big(\overline{X}, \texttt{OR}(\overline{Y})\big) = \texttt{OR}(\overline{X}, \overline{Y}) \tag{E_3}$$

$$\texttt{AND}\big(\overline{X}, \texttt{AND}(\overline{Y})\big) = \texttt{AND}(\overline{X}, \overline{Y}) \tag{E_4}$$

$$\texttt{SAND}\big(\overline{X}, \texttt{SAND}(\overline{Y}), \overline{Z}\big) = \texttt{SAND}(\overline{X}, \overline{Y}, \overline{Z}) \tag{$E_{4'}$}$$

$$\texttt{OR}(A) = A \tag{E_5}$$

$$\texttt{AND}(A) = A \tag{E_6}$$

$$\texttt{SAND}(A) = A \tag{$E_{6'}$}$$

$$\texttt{AND}\big(\overline{X}, \texttt{OR}(\overline{Y})\big) = \texttt{OR}\big(\texttt{AND}(\overline{X}, Y_1), \ldots, \texttt{AND}(\overline{X}, Y_\ell)\big) \tag{E_{10}}$$

$$\texttt{SAND}\big(\overline{X}, \texttt{OR}(\overline{Y}), \overline{Z}\big) = \texttt{OR}\big(\texttt{SAND}(\overline{X}, Y_1, \overline{Z}), \ldots, \texttt{SAND}(\overline{X}, Y_\ell, \overline{Z})\big) \tag{$E_{10'}$}$$

$$\texttt{OR}(A, A, \overline{X}) = \texttt{OR}(A, \overline{X}) \tag{E_{11}}$$

The numbering of the axioms in $E_{\mathcal{SP}}$ corresponds to the numbering of the axioms for the multiset semantics for standard attack trees, as presented in [9], while new axioms (involving SAND) are marked with primes.

Proof. The proof of this theorem follows the same line of reasoning as the proofs of Theorems 4.2 and 4.3 of Gischer [5], where series–parallel pomsets are axiomatized. The details can be found in the extended version of this work [6]. □

4.2 SAND Attack Trees in Canonical Form

Let $[\![\cdot]\!]$ be a semantics for (SAND) attack trees. A complete axiomatization of $[\![\cdot]\!]$ can be used to derive a canonical form of trees interpreted with $[\![\cdot]\!]$. Such canonical forms provide the most concise representation for equivalent trees and are the natural representatives of equivalence classes defined by $[\![\cdot]\!]$.

When SAND attack trees are interpreted using the SP semantics, their canonical forms consist of either a single basic action, or of a root node labeled with OR and subtrees with nested, alternating occurrences of AND and SAND nodes. Canonical forms correspond exactly to the sets of SP graphs labeled by $\mathbb{B}$ and they depict all attack alternatives in a straightforward way.

In the full version of this work [6], we show how to obtain canonical forms of SAND attack trees using the complete set of axioms $E_{\mathcal{SP}}$ for the SP semantics. By orienting the equations (E_3), (E_4), ($E_{4'}$), (E_5), (E_6), ($E_{6'}$), (E_{10}), ($E_{10'}$), and (E_{11}) from left to right, we obtain a term rewriting system, denoted by $R_{\mathcal{SP}}$. We show that $R_{\mathcal{SP}}$ is terminating and confluent. The canonical representations of SAND attack trees correspond to normal forms with respect to $R_{\mathcal{SP}}$. They are unique modulo commutativity of OR and AND.

Example 5. The canonical form of the SAND attack tree t in Figure 1 is the tree

$$t' = \texttt{OR}\big(\texttt{SAND}(\mathit{ftp}, \mathit{rsh}, \mathit{lobf}), \texttt{AND}(\mathit{ssh}, \mathit{rsa})\big)$$

shown in Figure 3. It is easily seen to be in normal form with respect to $R_{\mathcal{SP}}$.

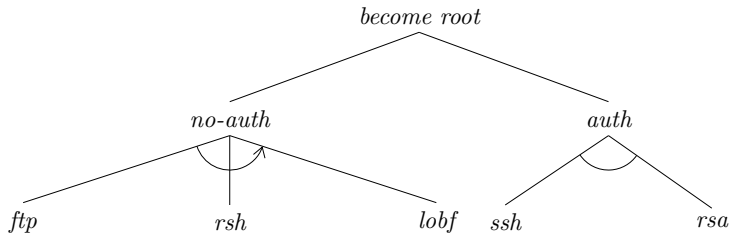

Fig. 3. SAND attack tree t' equivalent to SAND attack tree t from Figure 1

4.3 SP Semantics as a Generalization of the Multiset Semantics

Having a complete set of axioms for the SP semantics allows us to formalize the relation between SAND attack trees under the SP semantics and attack trees under the multiset semantics, denoted by $[\![\cdot]\!]_{\mathcal{M}}$. This is achieved by extracting a complete set of axioms for the multiset semantics for attack trees from the set $E_{\mathcal{SP}}$. Let $E_{\mathcal{M}}$ be the subset of axioms from $E_{\mathcal{SP}}$ that do not contain the SAND operator, i.e., $E_{\mathcal{M}} = \{(E_1), (E_2), (E_3), (E_4), (E_5), (E_6), (E_{10}), (E_{11})\}$.

Theorem 2. *The axiom system $E_{\mathcal{M}}$ is a complete set of axioms for the multiset semantics for attack trees.*

The proof can be found in a full version of this paper [6].

By comparing the complete sets of axioms $E_{\mathcal{SP}}$ and $E_{\mathcal{M}}$ we obtain that two attack trees are equivalent under the multiset semantics if and only if they are equivalent under the SP semantics. This is formalized in the following theorem.

Theorem 3. SAND *attack trees under the SP semantics are a* conservative extension *of attack trees under the multiset semantics.*

Proof. We need to prove that, for all standard attack trees t and t', we have $[\![t]\!]_{\mathcal{M}} = [\![t']\!]_{\mathcal{M}}$ if and only if $[\![t]\!]_{\mathcal{SP}} = [\![t']\!]_{\mathcal{SP}}$. From $E_{\mathcal{M}} \subset E_{\mathcal{SP}}$, we conclude that $[\![t]\!]_{\mathcal{M}} = [\![t']\!]_{\mathcal{M}}$ implies $[\![t]\!]_{\mathcal{SP}} = [\![t']\!]_{\mathcal{SP}}$. Conversely, we remark that the equations belonging to $E_{\mathcal{SP}} \setminus E_{\mathcal{M}}$ do not introduce new equalities on standard attack trees. A complete proof of this fact is given in the full version of this work [6]. □

5 Attributes

Attack trees do not only serve to represent security scenarios in a graphical way. They can also be used to quantify such scenarios with respect to a given parameter, called an *attribute*. Typical examples of attributes include the likelihood that the attacker's goal will be met and the minimal time or cost of an attack. Schneier described [19] an intuitive bottom-up algorithm for calculating attribute values of attack trees: attribute values are assigned to the leaf nodes and two

functions[6] (one for the OR and one for the AND refinement) are used to propagate the attribute value up to the root node. Mauw and Oostdijk showed [13] that if the binary operations induced by the two functions define a semiring, then the evaluation of the attribute on two attack trees equivalent with respect to the multiset semantics yields the same value. This result has been generalized to any semantics and attribute that satisfy a notion of *compatibility* [9] and we briefly discuss it for SAND attack trees at the end of this section. We start with a demonstration of how the bottom-up evaluation algorithm can naturally be extended to SAND attack trees.

An *attribute domain for an attribute* A_α *on* SAND *attack trees* is a tuple $D_\alpha = (V_\alpha, \triangledown_\alpha, \triangle_\alpha, \Diamond_\alpha)$, where V_α is a set of values and $\triangledown_\alpha, \triangle_\alpha, \Diamond_\alpha$ are families of k-ary functions of the form $V_\alpha \times \cdots \times V_\alpha \to V_\alpha$, associated to OR, AND, and SAND refinements, respectively. An *attribute for* SAND *attack trees* is a pair $A_\alpha = (D_\alpha, \beta_\alpha)$ formed by an attribute domain D_α and a function $\beta_\alpha : \mathbb{B} \to V_\alpha$, called *basic assignment* for A_α, which associates a value from V_α with each $b \in \mathbb{B}$.

Definition 6. *Let* $A_\alpha = \big((V_\alpha, \triangledown_\alpha, \triangle_\alpha, \Diamond_\alpha), \beta_\alpha\big)$ *be an attribute. The attribute evaluation function* $\alpha : \mathbb{T}_{\texttt{SAND}} \to V_\alpha$*, which calculates the value of attribute* A_α *for every* SAND *attack tree* $t \in \mathbb{T}_{\texttt{SAND}}$*, is defined recursively as follows.*

$$\alpha(t) = \begin{cases} \beta_\alpha(t) & \text{if } t = b,\ b \in \mathbb{B} \\ \triangledown_\alpha\big(\alpha(t_1), \ldots, \alpha(t_k)\big) & \text{if } t = \texttt{OR}(t_1, \ldots, t_k) \\ \triangle_\alpha\ \big(\alpha(t_1), \ldots, \alpha(t_k)\big) & \text{if } t = \texttt{AND}(t_1, \ldots, t_k) \\ \Diamond_\alpha\big(\alpha(t_1), \ldots, \alpha(t_k)\big) & \text{if } t = \texttt{SAND}(t_1, \ldots, t_k) \end{cases}$$

The following example illustrates the bottom-up evaluation of the attribute *minimal attack time* on the SAND attack tree given in Example 1.

Example 6. Let α denote the minimal time that the attacker needs to achieve her goal in the scenario of Example 1. We make the following assignments to the basic actions: *ftp* $\mapsto$ 3, *rsh* $\mapsto$ 5, *lobf* $\mapsto$ 7, *ssh* $\mapsto$ 8, *rsa* $\mapsto$ 9. Since we are interested in the minimal attack time, the function for an OR node is defined by $\triangledown_\alpha(x_1, \ldots, x_k) = \min\{x_1, \ldots, x_k\}$. The function for an AND node is $\triangle_\alpha\ (x_1, \ldots, x_k) = \max\{x_1, \ldots, x_k\}$, which models that the children of a conjunctively refined node are executed in parallel. Finally, in order to model that the children of a SAND node need to be executed sequentially, we let $\Diamond_\alpha(x_1, \ldots, x_k) = \sum_{i=1}^{k} x_i$. According to Definition 6, the minimal attack time is

$$\triangledown_\alpha\Big(\Diamond_\alpha\big(\Diamond_\alpha(3,5),7\big), \triangle_\alpha\ (8,9)\Big) = \min\Big(\Sigma\big(\Sigma(3,5),7\big), \max(8,9)\Big) = 9.$$

In the case of standard attack trees, the bottom-up procedure uses only two functions to propagate the attribute values to the root – one for conjunctive and one for disjunctive nodes. This means that the same function is employed to

[6] These are actually families of functions representing infinitely many k-ary function symbols, for all $k \geq 2$.

calculate the value of every conjunctively refined node, independently of whether its children need to be executed sequentially or can be executed simultaneously. Evidently, with SAND attack trees, we can apply different propagation functions for AND and SAND nodes, as in Example 6. Therefore, SAND attack trees can be evaluated over a larger set of attributes, and hence may provide more accurate evaluations of attack scenarios, than standard attack trees.

To guarantee that the evaluation of an attribute on equivalent attack trees yields the same value, the attribute domain must be *compatible* with a considered semantics [9]. Our complete set of axioms is a useful tool to check for compatibility with the SP semantics. Consider an attribute domain $D_\alpha = (V_\alpha, \triangledown_\alpha, \triangle_\alpha, \Diamond_\alpha)$, and let σ be a mapping $\sigma = \{\mathtt{OR} \mapsto \triangledown_\alpha, \mathtt{AND} \mapsto \triangle_\alpha, \mathtt{SAND} \mapsto \Diamond_\alpha\}$. Guaranteeing that D_α is compatible with a semantics axiomatized by E amounts to verifying that the equality $\sigma(l) = \sigma(r)$ holds in V_α, for every axiom $l = r \in E$. It is an easy exercise to show that the attribute domain for minimal attack time, considered in Example 6, is compatible with the SP semantics for SAND attack trees.

6 Conclusions

We have formalized the extension of attack trees with sequential conjunctive refinement, called SAND, and given a semantics to SAND attack trees in terms of sets of series-parallel graphs. This SP semantics naturally extends the multiset semantics for attack trees from [13]. We have shown that the notion of a complete set of axioms for a semantics and the bottom-up evaluation procedure can be generalized from attack trees to SAND attack trees, and have proposed a complete axiomatization of the SP semantics.

A number of recently proposed solutions focus on extending attack trees with defensive measures [18,9]. These extensions support reasoning about security scenarios involving two players – an attacker and a defender – and the interaction between them. In future work, we intend to add the SAND refinement to such trees. Afterwards, we plan to investigate sequential disjunctive refinement, as used for instance in [2]. Our goal is to propose a complete formalization of trees with attack and defense nodes, that have parallel and sequential, conjunctive and disjunctive refinements. Finally, our results will be used to extend the software application ADTool [8]. In particular, the axiomatization proposed in this paper and its term rewriting system $R_{\mathcal{SP}}$ will be implemented and used to decide on the equivalence of SAND attack trees.

Acknowledgments The research leading to these results has received funding from the European Union Seventh Framework Programme under grant agreement number 318003 (TREsPASS) and from the Fonds National de la Recherche Luxembourg under grant C13/IS/5809105.

References

1. Ardi, S., Byers, D., Shahmehri, N.: Towards a structured unified process for software security. In: SESS'06. pp. 3–10. ACM (2006)

2. Arnold, F., Hermanns, H., Pulungan, R., Stoelinga, M.: Time-Dependent Analysis of Attacks. In: Abadi, M., Kremer, S. (eds.) POST'14. LNCS, vol. 8414, pp. 285–305. Springer (2014)
3. Byers, D., Ardi, S., Shahmehri, N., Duma, C.: Modeling software vulnerabilities with vulnerability cause graphs. In: ICSM'06. pp. 411–422 (2006)
4. Camtepe, S., Yener, B.: Modeling and Detection of complex Attacks. In: SecureComm'07. pp. 234–243. IEEE (2007)
5. Gischer, J.L.: The Equational Theory of Pomsets. Theor. C. Sc. 61, 199–224 (1988)
6. Jhawar, R., Kordy, B., Mauw, S., Radomirović, S., Trujillo-Rasua, R.: Attack Trees with Sequential Conjunction. CoRR abs/1503.02261 (2015), `http://arxiv.org/abs/1503.02261`
7. Khand, P.A.: System level security modeling using attack trees. In: IC4'09. pp. 1–6 (2009)
8. Kordy, B., Kordy, P., Mauw, S., Schweitzer, P.: ADTool: Security Analysis with Attack–Defense Trees. In: Joshi, K.R., Siegle, M., Stoelinga, M., D'Argenio, P.R. (eds.) QEST'13. LNCS, vol. 8054, pp. 173–176. Springer (2013)
9. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack–Defense Trees. Journal of Logic and Computation 24(1), 55–87 (2014)
10. Kordy, B., Piètre-Cambacédès, L., Schweitzer, P.: DAG-Based Attack and Defense Modeling: Don't Miss the Forest for the Attack Trees. Computer Science Review 13–14(0), 1–38 (2014)
11. Kordy, B., Pouly, M., Schweitzer, P.: Computational Aspects of Attack–Defense Trees. In: Bouvry, P., Klopotek, M.A., Leprévost, F., Marciniak, M., Mykowiecka, A., Rybinski, H. (eds.) S&IIS'11. LNCS, vol. 7053, pp. 103–116. Springer (2011)
12. Kordy, B., Pouly, M., Schweitzer, P.: A Probabilistic Framework for Security Scenarios with Dependent Actions. In: Albert, E., Sekerinski, E. (eds.) iFM'14. LNCS, vol. 8739. Springer (2014)
13. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In: Won, D., Kim, S. (eds.) ICISC'05. LNCS, vol. 3935, pp. 186–198. Springer (2006)
14. Peine, H., Jawurek, M., Mandel, S.: Security Goal Indicator Trees: A Model of Software Features that Supports Efficient Security Inspection. In: HASE'08. pp. 9–18. IEEE Computer Society (2008)
15. Piètre-Cambacédès, L., Bouissou, M.: Beyond Attack Trees: Dynamic Security Modeling with Boolean Logic Driven Markov Processes (BDMP). In: EDCC'10. pp. 199–208. IEEE Computer Society, Los Alamitos, CA, USA (2010)
16. Pinchinat, S., Acher, M., Vojtisek, D.: Towards Synthesis of Attack Trees for Supporting Computer-Aided Risk Analysis. In: Software Engineering and Formal Methods. LNCS, vol. 8938, pp. 363–375. Springer (2014)
17. Qin, X., Lee, W.: Attack plan recognition and prediction using causal networks. In: 20th Annual Computer Security Applications Conference. pp. 370–379 (2004)
18. Roy, A., Kim, D.S., Trivedi, K.S.: Attack Countermeasure Trees (ACT): towards unifying the constructs of attack and defense trees. Security and Communication Networks 5(8), 929–943 (2012)
19. Schneier, B.: Attack Trees: Modeling Security Threats. Dr. Dobb's Journal of Software Tools 24(12), 21–29 (1999)
20. Wang, J., Whitley, J.N., Phan, R.C.W., Parish, D.J.: Unified Parametrizable Attack Tree. Int. Journal for Information Security Research 1(1), 20–26 (2011)
21. Wen-ping, L., Wei-min, L.: Space Based Information System Security Risk Evaluation Based on Improved Attack Trees. In: (MINES'11). pp. 480–483 (2011)
22. Willemson, J., Jürgenson, A.: Serial Model for Attack Tree Computations. In: Lee, D., Hong, S. (eds.) ICISC'09. LNCS, vol. 5984, pp. 118–128. Springer (2010)